

CLAIMS

We claim:

1. A graphics processor, comprising:

a multithreading, multi-core graphics engine to process pixel data;

a render-cache, readily accessible to the graphics engine, to store pixel data; and,

a render-cache controller to maintain the order in which threads are dispatched to the graphics engine, and to maintain data coherency between the render-cache and a main memory.

2. The graphics processor of claim 1, further including:

raster logic to generate threads, each thread including at least one cache-line address indicating the location of pixel data in the render-cache; and,

a thread dispatcher to dispatch each thread to the graphics engine only when the render-cache controller indicates that the at least one cache-line address is valid.

3. The graphics processor of claim 2, wherein the multithreading, multi-core graphics engine is to process pixel data for rendering 3D graphics.

4. The graphics processor of claim 2, wherein the at least one cache-line address is valid if the render-cache controller indicates a cache hit during a look-up

operation, and the pixel data stored at the at least one cache-line address is not in-flight.

5. The graphics processor of claim 2, wherein the render-cache controller is to block a thread from dispatching to the graphics engine if the thread specifies a cache-line address of the render-cache containing a pixel in flight.

6. The graphics processor of claim 2, wherein the render-cache controller comprises:

a content addressable memory to map pixel coordinates to a cache-line address of the render-cache, the address specifying a location in the render-cache where pixel data corresponding to the pixel coordinates is stored;

a pixel mask array having a bit for every entry of the content addressable memory, each bit to indicate whether previously allocated pixel data is in flight; and,

a cache-line status array with a bit for every entry of the content addressable memory, each bit to indicate the availability of a cache-line in the render-cache.

7. The graphics processor of claim 6, wherein the render-cache controller further comprises a pipeline interface to receive cache-line addresses when the graphics engine reads from or writes to the render-cache.

8. The graphics processor of claim 7, wherein the render-cache controller maintains the in-flight status of pixel data stored in the render-cache by receiving cache-line addresses from the pipeline interface.
9. The graphics processor of claim 8, wherein the render-cache controller changes the status of pixel data stored at a particular cache-line address to indicate that the pixel data is in-flight when the render-cache controller receives the address of the cache-line via the pipeline when the graphics engine reads the pixel data from the cache-line associated with the cache-line address.
10. The graphics processor of claim 8, wherein the render-cache controller changes the status of pixel data stored at a particular cache-line address to indicate that the pixel data is no longer in flight when the render-cache controller receives the address of the cache-line via the pipeline when the graphics engine writes the pixel data to the cache-line associated with the cache-line address.
11. A render-cache controller comprising:
 - a content addressable memory to map pixel coordinates to a cache-line address of a render-cache, the cache-line address specifying a location in the render-cache where pixel data corresponding to the pixel coordinates is stored;
 - a pixel mask array having a bit for every entry of the content addressable memory, each bit to indicate whether previously allocated pixel data is in flight;
 - and,

a cache-line status array with a bit for every entry of the content addressable memory, each bit to indicate the availability of a cache-line in the render-cache.

12. The render-cache controller of claim 11, furthering comprising:

a pipeline interface to receive a cache-line address when a graphics engine reads or writes pixel data to the render-cache.

13. The render-cache controller of claim 12, wherein the pixel mask array is to set a bit corresponding with a cache-line address of the render-cache when the pixel data stored at the cache-line address is read by the graphics engine and the cache-line address is received by the render-cache controller via the pipeline interface, the set bit indicating that the pixel data read from the cache-line address is in flight.

14. The render-cache controller of claim 12, wherein the pixel mask array is to reset a bit corresponding with a cache-line address of the render-cache when pixel data is written by the graphics engine to the cache-line address and the cache-line address is received by the render-cache controller via the pipeline interface, the reset bit indicating that the pixel data written to the cache-line address is not in flight.

15. The render-cache controller of claim 12, wherein the content addressable memory blocks the thread dispatcher from dispatching threads generated by

raster logic if the threads include cache-line addresses of the render-cache containing pixel data in flight.

16. The render-cache controller of claim 12, wherein the pixel mask array indicates whether cache-line addresses included in the thread are associated with pixel data in flight.

17. The render-cache controller of claim 12, wherein pixel data is in flight if it has been read by the graphics engine more recently than it has been written to the render-cache.

18. A method to pre-allocate pixel data to a render-cache, the method comprising:

- checking a content addressable memory to determine whether pixel data for a particular pixel has been previously allocated to the render-cache;

- if the pixel data for the particular pixel has not been previously allocated to the render-cache then

- checking a cache-line status array to determine an address of an available cache-line in the render-cache,

- evicting pixel data from the address of the available cache-line,

- writing the pixel data to the address of the available cache-line in the render-cache, and

- setting a bit in a pixel mask array to indicate that the pixel data written to the address of the available cache-line is in flight; and

if the pixel data for the particular pixel has been previously allocated to the render-cache then

checking a pixel mask array to determine whether the previously allocated pixel data is in flight,

stalling, if the previously allocated pixel data is in flight, and

dispatching a thread to the graphics engine if the previously allocated pixel data is not in-flight.

19. The method of claim 18, wherein checking the content addressable memory to determine whether pixel data for a particular pixel has been previously allocated to the render-cache includes comparing the X and Y coordinates of the particular pixel to X and Y coordinates of pixel data stored in the content addressable memory and determining that the pixel data has been previously allocated if the comparison results in a match.

20. The method of claim 18, wherein checking a cache-line status array to determine an address of an available cache-line in the render-cache includes selecting an available cache-line based on a cache-line selection algorithm.

21. The method of claim 20, wherein the cache-line selection algorithm is based on a least recently used selection algorithm.

22. The method of claim 18, wherein evicting pixel data from the address of the available cache-line includes writing the pixel data to a memory.

23. A portable media device comprising:

a CPU;

a main memory;

a graphics processor, the graphics processor comprising a multithreading, multi-core graphics engine to generate graphics by processing pixel data, a render-cache to store pixel data, and a render-cache controller to maintain the order in which threads are dispatched to the graphics engine and to maintain data coherency between the render-cache and the main memory; and

a battery to provide power to the CPU, the main memory, and the graphics processor.

24. The portable media player of claim 23, further comprising:

a liquid crystal display to display the graphics generated by the graphics processor.

25. The portable media player of claim 23, wherein the graphics processor further comprises a raster logic to generate threads, each thread including at least one cache-line address indicating the location of pixel data in the render-cache, and a thread dispatcher to dispatch each thread to the graphics engine only when the render-cache controller indicates that the at least one cache-line address is valid.

26. The portable media player of claim 25, wherein the at least one cache-line address is valid if the render-cache controller indicates a cache hit during a lookup operation, and the pixel data stored at the at least one cache-line address is not in-flight.

27. The portable media player of claim 26, wherein the graphics engine is to generate 3D graphics.